Tarea 019 – Javascript – clase Bola (IV) – Herencia Pelota - rebote paredes

Los resultados de todas las tareas incorporarán, además del código fuente, los comentarios precisos y necesarios para su fácil compresión. No escatimes esfuerzos en comentar el código, es una buena práctica para aprender, además de ser muy útil para modificaciones o reutilizaciones futuras.

Sería muy buena práctica añadir el título y el enunciado del ejercicio (como comentarios) al principio del código fuente.

Añade documentos en formato Word con capturas de la salida por pantalla (al ejecutar la página) si consideras que queda más clara la resolución del ejercicio.

Deberás entregar dos versiones del resultado:

1. **Un único documento en Word sin comprimir**: que contenga todas las líneas de código, y por orden, primero el código **html**, luego el código **css** y por último el código **js**. Este archivo servirá para que el profesor haga anotaciones y/o correcciones a los alumnos.
2. **Los archivos de código fuente**: en un único archivo, ya sea de extensión: **html**, **js**, o **css**. Si precisas entregar varios archivos, comprímelos en un único **zip**. (No admito rar).

El nombre del archivo entregado comenzará por tu nombre seguido por TareaXXX. Ejemplo: **federicoTarea014.zip**

019 – clase Bola (IV) – Herencia Pelota – rebote paredes

Enunciado: partiendo de la tarea anterior que utiliza la clase Bola, se pide:

1. Crear una clase Pelota, que hereda de Bola, para que cada pelota tenga, además:
   - Su propia velocidad en el eje X (propiedad vx) que puede ser positiva o negativa. Positiva cuando avance de izquierda a derecha, y negativa cuando sea en dirección contraria.
   - Su propia velocidad en el eje Y (propiedad vy) que puede ser positiva o negativa. Positiva cuando avance de arriba a abajo, y negativa cuando sea en dirección contraria.
   - Un método para comprobar que va a chocar con las paredes, e invertir las velocidades justo antes de que ocurra (cambiando de signo las velocidades).
   - Un método que mire a ver si cada pelota va a chocar con el resto de pelotas (no consigo misma).
2. Cambia el botón “Booola” por “Pelopelopta”, de tal forma que, cuando se haga clic en este botón se creará una nueva pelota con radio, posición, colores y velocidades aleatorias.
3. Cada nuevo objeto Pelota se almacenará en un array con scope global llamado **arrayPelotas**. Evita que este array sea estático a cualquier clase, así evitamos que haya un acoplamiento fuerte entre la clase y el array (sería más difícil trabajar con el array).

Otras sugerencias:

Podemos resolver el movimiento de las pelotas de dos formas:

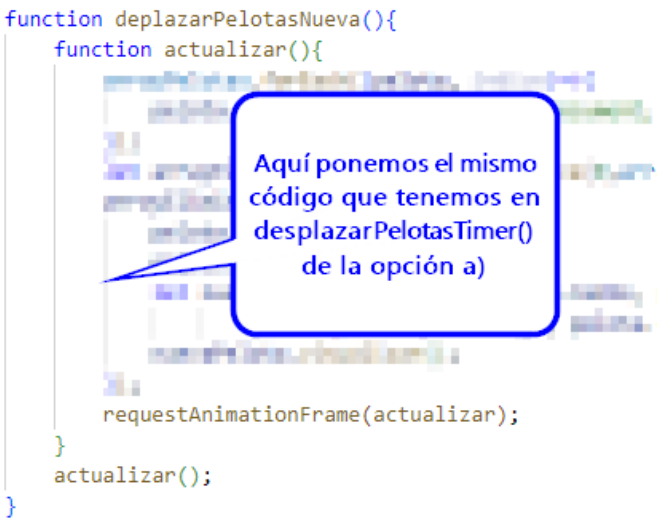
a) **Utilizando SetInterval()**. En este caso se podrá parar con la tecla “Fin” o de cualquier otra forma. Se recomienda llamar a la función cada 20 ms.

```
function iniciarMovimiento(){
    intervaloTimer=setInterval(deplazarPelotasTimer, 20);

}
```

b) Utilizando `requestAnimationFrame`

El código que pudiéramos tener en desplazarPelotasTimer() de la opción a), podemos envolverlo de la siguiente forma:



Con la opción b) no se utiliza setInterval()